Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 8 (2012) 232 – 238

Procedia Computer Science

New Challenges in Systems Engineering and Architecting
Conference on Systems Engineering Research (CSER)
2012 – St. Louis, MO
Cihan H. Dagli, Editor in Chief
Organized by Missouri University of Science and Technology

Flight test results for UAVs using boid guidance algorithms

Jason B. Clark, David R. Jacques

Air Force Institute of Technology, 2950 Hobson Way, Wright Patterson AFB 45433, USA

Abstract

A critical technology for operating groups of Uninhabited Aerial Vehicles (UAVs) is distributed guidance. Distributed guidance allows an operator to command several vehicles at the same time, reduces operator workload, and adds redundancy to the system. Some of the leading software candidates for achieving distributed guidance are known as Boid Guidance Algorithms (BGAs), which are agent-based techniques relying on the interactions of simple behaviors.

Flight tests are crucial to the advancement of flight technologies such as BGAs, and this was identified as an important area for development. This paper presents the results from the 2005 flight tests of BGAs at NASA Dryden Flight Research Center with two RnR Products' APV-3 UAVs employing CloudCap Technology's Piccolo autopilot system.

Major challenges in these flight tests include the use of a waypoint-following system, limited computation resources, and management of safety procedures. The conclusions of this work include the need for using a path-following platform and completion of a full system optimization. This work is an important step in the development of a deployable distributed guidance system.

Keywords: Agents, swarm guidance, multi-vehicle control, boid

1. Introduction

The ability for a single operator to manage multiple aircraft is a sought-after goal. Possible benefits of this technology include improved operator efficiency, the opportunity for distributed function, redundancy in hardware, reduced production and operating costs, and improved safety. However, the management of several aircraft simultaneously is wrought with challenges.

1877-0509 © 2012 Published by Elsevier B.V.
doi:10.1016/j.procs.2012.01.048

This research stems from a collaborative project of NASA Dryden Flight Research Center (DFRC) and Ames Research Center (ARC) entitled Networked UAV Teams (NUAVT) [6]. The focus of the project was to show the feasibility of using cooperative UAVs to assist in forest fire fighting activities. However, the technologies developed are general enough for use in many different applications.

Previous approaches to this problem are computationally expensive and incur a significant scaling penalty with regard to group size [9][10]. Since small UAVs typically have limited computing capacity, this problem is of high importance. BGAs have shown a great deal of success in coordinating large numbers of simulated aircraft [4][13], but there are few known flight tests.

2. Boid Guidance Algorithms

Boid Guidance Algorithms (where "Boid" stands for "Bird android") are rule-based guidance methods inspired from observations of animal flocks and swarms [12]. It was proposed that these complex emergent behaviors could be explained if each animal agent were to follow a set of very simple rules as shown in Fig. 1 [11]. The combination of these rules can lead to seemingly intelligent behavior.

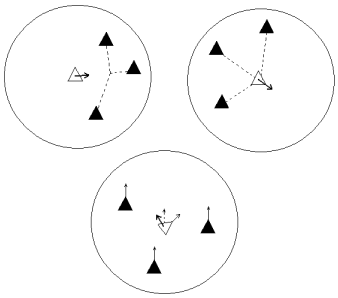

Fig.1. Clockwise from left: Flocking behavior, collision avoidance behavior, and heading matching behavior.

The behavioral rules are combined by weighting the acceleration command of each behavior and summing. Determining the weightings is typically the most challenging aspect of working with BGAs.

Previous work on BGAs involved exponentially scaling the behavior weightings [4]. That is, as one boid approached another, its collision avoidance behavior would get a higher weighting. The inverse would occur as the boid moved farther from its neighbors. This research did not use such a method as it incurs a greater on-line computation penalty yet presents similar optimization challenges. Instead, this implementation used a look-up table scheduled by a contingency management system.

BGAs were chosen for this work due to their low computation requirements, but were further modified so that most computation takes place off-line. This modification reduces the workload on the flight system. In addition, this system has a limited scaling penalty.

The rules used here are velocity matching (1), flock centering (2), collision avoidance (3), target seeking (4), and obstacle avoidance (5), where $\vec{x}$, $\vec{v}$, and $\vec{a}_{(\cdot)}$ are the current boid position, velocity, and partial acceleration, respectively.

$$\vec{a}_{match} = \frac{\vec{v}_{nearest}}{\|\vec{v}_{nearest} - \vec{v}\|} \quad (1)$$

$$\vec{a}_{flock} = \frac{\vec{x}_{center} - \vec{x}}{\|\vec{x}_{center} - \vec{x}\|} \text{ , where } \vec{x}_{center} = \frac{1}{n}\sum_{i=0}^{n} \vec{x}_i \quad (2)$$

$$\vec{a}_{col} = -\frac{\vec{x}_{nearest} - \vec{x}}{\|\vec{x}_{nearest} - \vec{x}\|} \quad (3)$$

$$\vec{a}_{seek} = \frac{\vec{x}_{waypoint} - \vec{x}}{\|\vec{x}_{waypoint} - \vec{x}\|} \quad (4)$$

$$\vec{a}_{obs} = -\frac{\vec{x}_{obstacle} - \vec{x}}{\|\vec{x}_{obstacle} - \vec{x}\|} \quad (5)$$

2.1. Contingency Management System

Much like a linearized control system, using only one set of behavior weightings is valid for only a certain range of conditions. For a robust system, the weightings must change in response to changes in the environment.

To accomplish this, trigger flags were defined to indicate the state of the environment. For instance, if an aircraft passes within a specified range of its neighbor, the value of a flag is changed. Here, 48 different flag combinations were identified. For each combination, a set of weightings is specified.

For all behaviors except obstacle avoidance, nominal conditions are indicated by a "Level 2" flag. For instance, if the aircraft are nearby each other without being too close, headed in about the same direction, and have reached the target area, then the flocking, collision avoidance, heading matching, and target seeking behavior flags will all be Level 2. The obstacle avoidance behavior flag has three levels: Level 3 for nominal, Level 2 if the aircraft is heading toward an obstacle, and Level 1 if a collision is imminent.

The parameters that trigger the flags (such as allowable distance between aircraft) must be determined by the user based upon aircraft performance characteristics. For an aircraft model with significant uncertainty, the parameters may be set arbitrarily restrictive at first, but eased as the aircraft model is refined.

2.2. Other modifications

In addition to the contingency management system, other changes to the basic algorithm were required. Commands generated by the BGA were limited according to speed and turn magnitude, so as to reflect the limits of the control system. The turn limiting was based on a maximum load factor constrained by either the drag and minimum thrust or the stall limit [1].

To ensure the aircraft stayed within the prescribed boundaries for flight tests, when an aircraft came within a certain distance of the boundaries, it was given full power in the direction orthogonal to the boundary. The concern with this approach is that full priority is given to area containment at the expense of collision avoidance. However, the safety of the public and operators is paramount.

In addition to limiting the commands, some smaller modifications to the algorithm were required. Since the number of waypoints available to each aircraft was limited (see "Challenges" section), the commanded paths were modified to remove waypoints on relatively long and straight sections. Given the limited accuracy of the UAV guidance system, this modification had negligible effect on UAV behavior.

Further details on these modifications and the bases for calculations are found in Clark [3].

3. Optimization

The weightings for the BGA were optimized using a Simple Genetic Algorithm (SGA) [8]. Previous work showed success using SGAs for optimization of complex functions and BGAs in particular [7][5].

SGAs use an evolutionary approach to optimize according to an objective function. Given an initial population of binary strings, an SGA will evaluate the fitness of each individual string. Then, using the processes of reproduction, crossover, and mutation, the SGA produces a new population of strings. With each successive generation, the probability for an improvement of overall fitness generally increases.

This research used a SGA developed by the University of Sheffield for use in Matlab. It was implemented for this work with a population of 20, a generation gap of 0.9, a bit precision of 8, and a variable range of 0 to 100 for the five behavior weightings. The SGA ran for 500 generations after which the top performing individual was selected for inclusion in the BGA.

The objective function consisted of a sum of the "Level 1" flags corresponding to each behavior, with an additional weight of 4 on the target seeking behavior flag. A Level 1 flag in either of the anti-collision behaviors resulted in an artificially large objective value of 5000, essentially taking it out of future populations. Thus, a weighting set which gets the flock to the target location the fastest with no collisions, while minimizing the times the aircraft drift out of tolerance on the cohesion parameters, has the greatest chance of selection.

Due to lack of computing resources for timely completion, only one set of behavior weightings was selected for optimization, with three others being set manually. The full contingency management system desired would include at least 48 such sets. Therefore, the obtained set was optimized for a specific set of conditions, and deviations from these nominal conditions can result in undesired performance results.

4. Challenges

There were several challenges to testing the BGA system, both self-imposed and exogenous. Perhaps the most serious impediment to achieving an ideal test of the BGAs was the use of an autopilot system employing waypoint-following logic. While suitable for many applications, waypoint-following is generally inadequate for robust collision avoidance. Since collision avoidance is a key component of BGAs, the waypoint-following logic presented a major difficulty in ensuring test success.

In addition, the NASA ARC-developed interface program could accept only 15 waypoints per path for each aircraft, leading to larger distances between waypoints. Aircraft motion between the waypoints is not dictated by the BGA, thus larger waypoint separation results in an increased chance of collision.

A further constraint of the selected autopilot system was the requirement that the final waypoint either link back to the first point or be made an orbit point. To increase certainty in the aircraft positions, the final point in all paths was made an orbit point. The challenge is found in that the aircraft starts entering the orbit as soon as it crosses the next-to-last waypoint. This can lead to significant deviation from the path commanded by the BGA.

As mentioned earlier, the implemented BGA is minimally optimized. This constraint limited the tolerance of the algorithm to deviations from its optimized condition. It was therefore important to choose an optimization case that was a more challenging environment than expected in flight.

Some self-imposed safety features also presented challenges to performing a robust test of the algorithm. In order to ensure flight safety, the flight paths generated by the algorithm were inspected by the operators before the aircraft were allowed to follow them. This created two difficulties: the paths were not updated in flight, resulting in open-loop guidance; and accurate initial conditions were not available to the algorithm.

5. Flight test results

Flight tests were conducted by NASA DFRC and ARC in January 2005 at Edwards Air Force Base. The test aircraft were RnR Products' APV-3 vehicles, with wing span of 12 feed, empty weight of 30 lbs, maximum speed of 90 mph, minimum speed of 45 mph, and a 7.5 hp gasoline engine.

The aircraft transited the test area from corner to corner (for reference, the long horizontal edge is ~2 miles), while avoiding virtual obstacles and one another. Fig. 2(a) shows results from one such test, with diamonds and squares as the waypoints for the first and second aircraft, respectively.

In this scenario, the aircraft are travelling from lower left to upper right with no obstacles. The difficulty of establishing achievable initial conditions is visible as is the entry into orbit of the last

waypoint. With the exception of the endpoints, the flight path meets the goal of coordinated group flight without collision or departure from the test area. In Fig. 2(b), the selected aircraft initial conditions correspond well to the actual path entry points. However, due to the distance between the final orbit point and the next-to-last waypoint, there is significant departure from the desired path. This illustrates the need for the next-to-last waypoint to be near the final orbit point when using a waypoint-guidance system.

In Fig. 3(a), the aircraft avoid simulated obstacles, although the path deviation of the northernmost aircraft nearly crosses one of the obstacle boundaries. Again, this is a result of the next-to-last waypoint being too far from the final orbit point. The aircraft safely navigate among four simulated obstacles in Fig. 3(b). The distance between waypoints leads the aircraft to approach an obstacle closer than desired.

Fig. 4(a) shows a clear violation of a simulated obstacle. This failure is due to the reduced number of available waypoints for the path. None of the waypoints are within the obstacle boundaries, so it was not an algorithm failure. The results shown in Figure 4(b) are very similar to those of Figure 4(a). The obstacle boundary violation is again due to the reduced number of available waypoints.

Figure 5 demonstrates a failure of the algorithm, as waypoints were placed within the obstacle boundaries. This is due to the use of a minimal set of schedulable behavior weightings. If the full set were available, the contingency management system may have prevented the incursion. Also of note is the excursion of the aircraft with the dotted path early in the flight. This is another artifact of the waypoint-following algorithm used by the autopilot combined with inaccurate initial conditions selection.

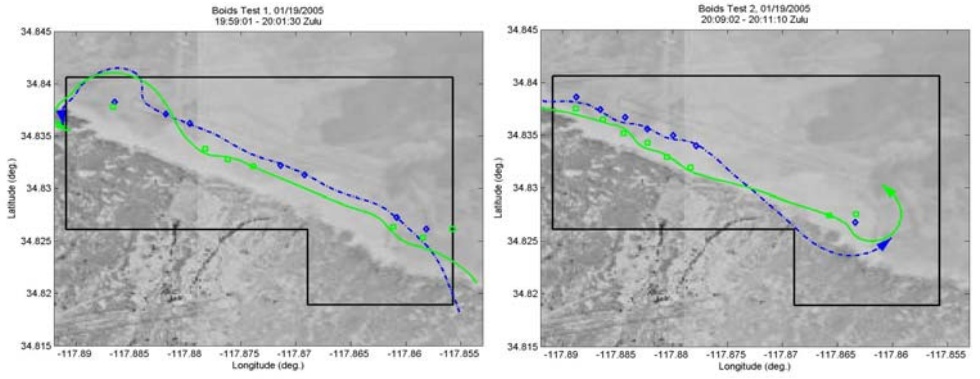

Fig. 2. (a) BGA Test Scenario 1; (b) BGA Test Scenario 2

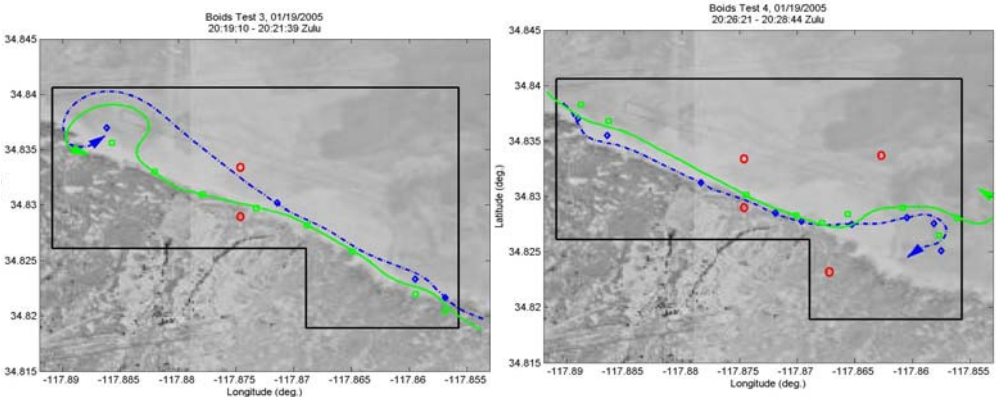

Fig. 3. (a) BGA Scenario 3; (b) BGA Scenario 4

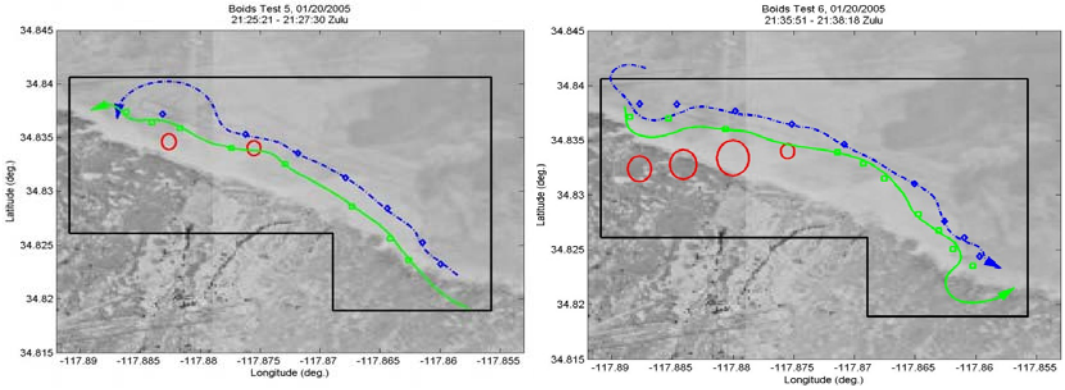

Fig. 4. (a) BGA Scenario 5; (b) BGA Scenario 6

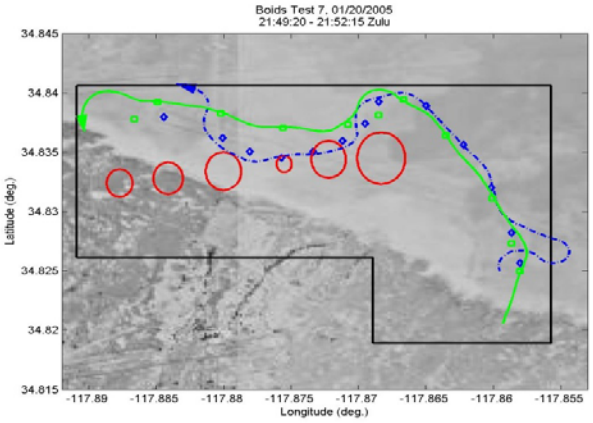

Fig. 5. BGA Scenario 7

6. Conclusions and Future Work

Although more work is needed to develop a robust system, these flight tests have demonstrated an emerging capability for safely and efficiently UAVs utilizing boid algorithms. Future work should include a path-following system for robust collision avoidance and dynamic path development. On-aircraft computation and calculating a complete set of weightings would improve results. It would also be instructive to use "self-optimizing" techniques such as Particle Swarm Optimization [2], which allows for adaptability by finding an optimum balance between exploration and exploitation. Tests exploring other emergent behaviors, such as orbiting or drag reduction [3], would be good next steps.

References

[1] Anderson, J.D. Airplane Performance: Accelerated Flight. *Aircraft Performance and Design.* Boston: McGraw-Hill; 1999.

[2] Banks, A., Vincent, J. and Phalp, K. Particle swarm guidance for autonomous unmanned aerial vehicles in an air defence role. *Journal of Navigation* 2008: **61**; 9-19.

[3] Clark, J. B. Decentralized UAV Guidance Using Modified Boid Algorithms. Ames, IA: Iowa State University Press; 2004.

[4] Crowther, W. Rule-based guidance for flight vehicle flocking. *I MECH E Pt G Jrnl of Aero Eng*, Apr 2004; ***218***(2): 111-124.

[5] Dimock, G. and Selig, M. The Aerodynamic Benefits of Self-Organization in Bird Flocks. *41st Aerospace Sciences Meeting and Exhibit*. Reno, NV; January 2003.

[6] Flinn, E. Testing for the 'boids'. *Aerospace America*. June 2005.

[7] Goldberg, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley; 1989.

[8] Holland, J. *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: The University of Michigan Press; 1975.

[9] How, J., King, E. and Kuwata, Y. Flight Demonstrations of Cooperative Control for UAV Teams. *AIAA 3rd Unmanned Unlimited Technical Conference, Workshop and Exhibit*. Chicago; September 2004.

[10] Inalhan, G., Stipanovic, D., and Tomlin, C. Decentralized Optimization with Application to Multiple Aircraft Coordination. *4th International Conference on Cooperative Control and Optimization*. Destin, FL; November 2003.

[11] Park, C., Tahk, M., and Bang, H. Multiple Aerial Vehicle Formation Using Swarm Intelligence. *AIAA Guidance, Navigation, and Control Conference and Exhibit*. Austin, TX; August 2003.

[12] Reynolds, C. Flocks, Herds, and Schools: A Distributed Behavioral Model. *Computer Graphics* 1987; ***21***(4), 25-34.

[13] Watson, N., John, N., and Crowther, W. Simulation of Unmanned Air Vehicle Flocking. *IEEE Theory and Practice of Computer Graphics Conference*. Birmingham, UK; June 2003.